Testing Methods

Unit Testing Methods

Unit testing focuses on validating individual components of the system in isolation. This ensures that specific functions, such as authentication, attendance logging, and device registration, perform as intended before system-wide integration.

Tools Used:

- Jest: A JavaScript testing framework used to write and run unit tests for core system functions.
- Postman: Used in conjunction with mock data and the REST API to test backend functionality and API endpoints.

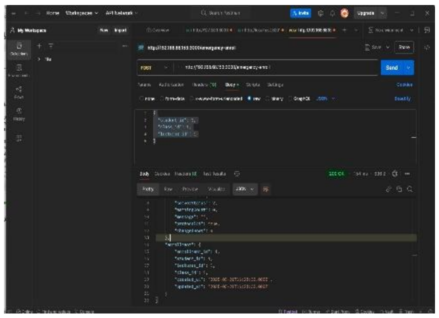
Examples of Unit-Tested Functions:

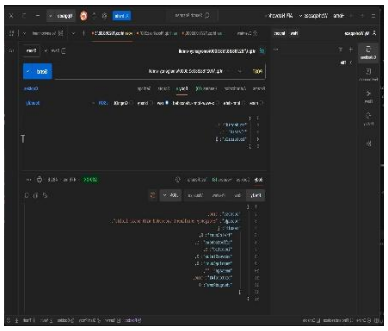
- validateUserLogin() – Verifies that users can authenticate with correct credentials and are denied with invalid ones.
- registerDevice(deviceId) – Ensures that only unique, authorized devices are registered and tracked.
- recordAttendance(studentId, timestamp) – Checks that attendance records are created accurately with required metadata.
- capturePhoto() – Tests the function that stores a student's image file and associates it with their ID and course record.

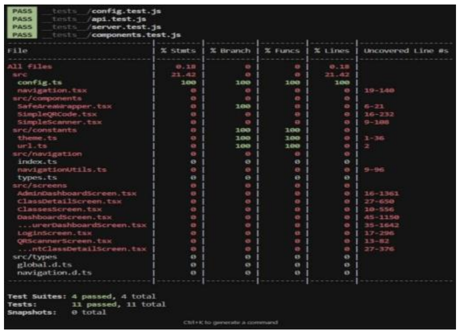
5.2 Test-Driven Development (TDD) Evidence

Test-Driven Development (TDD) was followed during critical feature implementation. TDD is a software development approach where tests are written **before** the actual code is developed.

Screenshots of Sucessful Testing:







```
PASS  __tests__/config.test.js
PASS  __tests__/api.test.js
PASS  __tests__/server.test.js
PASS  __tests__/components.test.js
-------------------------------|---------|----------|---------|---------|-------------------
File                           | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
-------------------------------|---------|----------|---------|---------|-------------------
All files                      |    0.18 |        0 |       0 |    0.18 |
 src                           |   21.42 |        0 |       0 |   21.42 |
  config.ts                    |     100 |      100 |     100 |     100 |
  navigation.tsx               |       0 |        0 |       0 |       0 | 19-140
 src/components                |       0 |        0 |       0 |       0 |
  SafeAreaWrapper.tsx          |       0 |      100 |       0 |       0 | 6-21
  SimpleQRCode.tsx             |       0 |        0 |       0 |       0 | 16-232
  SimpleScanner.tsx            |       0 |        0 |       0 |       0 | 9-108
 src/constants                 |       0 |      100 |     100 |       0 |
  theme.ts                     |       0 |      100 |     100 |       0 | 1-36
  url.ts                       |       0 |      100 |     100 |       0 | 2
 src/navigation                |       0 |        0 |       0 |       0 |
  index.ts                     |       0 |        0 |       0 |       0 |
  navigationUtils.ts           |       0 |        0 |       0 |       0 | 9-96
  types.ts                     |       0 |        0 |       0 |       0 |
 src/screens                   |       0 |        0 |       0 |       0 |
  AdminDashboardScreen.tsx     |       0 |        0 |       0 |       0 | 16-1361
  ClassDetailScreen.tsx        |       0 |        0 |       0 |       0 | 27-650
  ClassesScreen.tsx            |       0 |        0 |       0 |       0 | 10-556
  DashboardScreen.tsx          |       0 |        0 |       0 |       0 | 45-1150
  ...urerDashboardScreen.tsx   |       0 |        0 |       0 |       0 | 35-1642
  LoginScreen.tsx              |       0 |        0 |       0 |       0 | 17-296
  QRScannerScreen.tsx          |       0 |        0 |       0 |       0 | 13-82
  ...ntClassDetailScreen.tsx   |       0 |        0 |       0 |       0 | 27-376
 src/types                     |       0 |        0 |       0 |       0 |
  global.d.ts                  |       0 |        0 |       0 |       0 |
  navigation.d.ts              |       0 |        0 |       0 |       0 |
-------------------------------|---------|----------|---------|---------|-------------------

Test Suites: 4 passed, 4 total
Tests:       11 passed, 11 total
Snapshots:   0 total
                    Ctrl+K to generate a command
```

5.3 Functional Testing

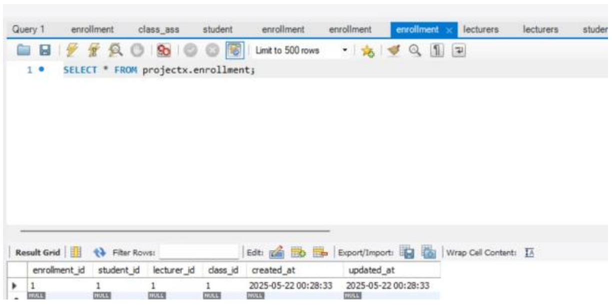
Functional testing was conducted to verify that each system feature meets its specified requirements. These tests were performed from the user’s perspective, focusing on input/output behavior.

Key functional tests:

- Admin login, user creation, and course management.
- Lecturer device registration and manual attendance override.
- Student identification via photo and accurate record creation.

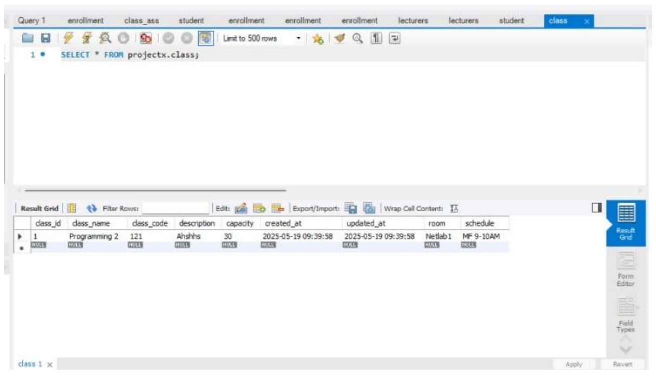
Test Cases Included:

- “Lecturer successfully records student attendance.”
- “System prevents unregistered device access.”



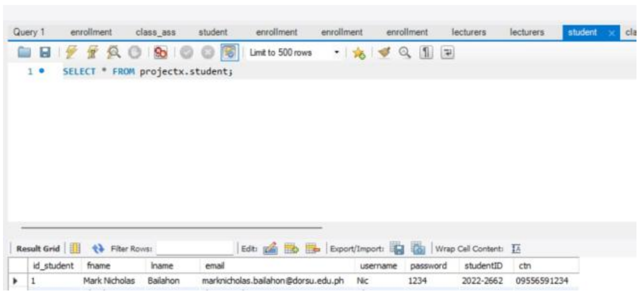
```
SELECT * FROM projectx.enrollment;
```

enrollment_id	student_id	lecturer_id	class_id	created_at	updated_at
1	1	1	1	2025-05-22 00:28:33	2025-05-22 00:28:33
NULL	NULL	NULL	NULL	NULL	NULL



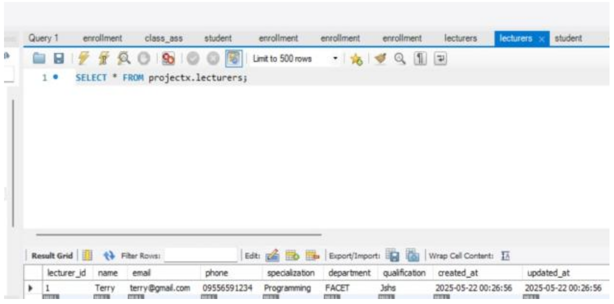
```
SELECT * FROM projectx.class;
```

class_id	class_name	class_code	description	capacity	created_at	updated_at	room	schedule
1	Programming 2	121	Ahshhs	30	2025-05-19 09:39:58	2025-05-19 09:39:58	Netlab1	MF 9-10AM
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



```
SELECT * FROM projectx.student;
```

id_student	fname	lname	email	username	password	studentID	ctn
1	Mark Nicholas	Bailahon	marknicholas.bailahon@dorsu.edu.ph	Nic	1234	2022-2662	09556591234



```
SELECT * FROM projectx.lecturers;
```

lecturer_id	name	email	phone	specialization	department	qualification	created_at	updated_at
1	Terry	terry@gmail.com	09556591234	Programming	FACET	Jshs	2025-05-22 00:26:56	2025-05-22 00:26:56